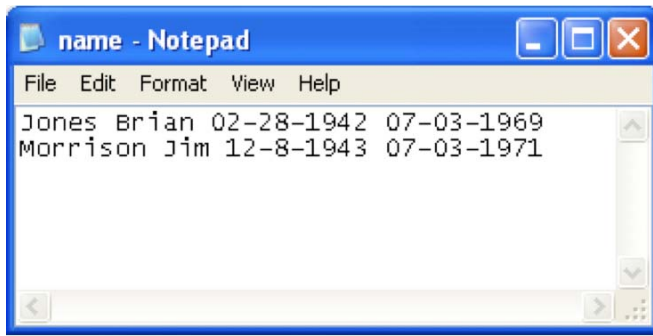


Lab 4 – Input\Output in VB Using A Data File

Introduction

You may want to read some data from an input file and write results into another output file. In these cases, it is useful to use a plain “text file” (“*.txt”) which includes no information other than plain text. Files for other software (e.g., MS Word) contain complicated data that are used to format the information in the file.

Below is a file containing the names of a few dead rock stars and the dates they died. You may not have realized that Brian Jones (original rhythm guitarist for the Rolling Stones) and Jim Morrison died two years apart on the same day.

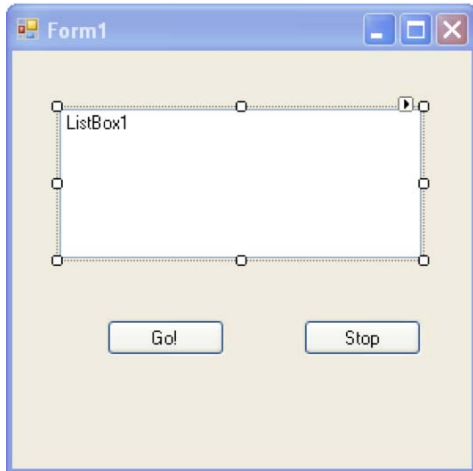


Access Visual Basic

1. Open VB from the “start” menu, that is:
Start\All Programs\Microsoft Visual Basic 2008 Express Edition
2. Select “**File → New Project**” to create a new project
3. Select “**Windows Form Application**”. You can give your “application” a name, or simply accept the default offered “**WindowsApplication1**”. Click “**OK**”.

Prepare the Graphic User Interface

Develop the "form" shown in the figure above that we will use in the project. The form should contain a label, a text box, two command buttons, and a picture box with the following properties, and look like:



Add VB Code to the Project

Let's add some code to the "Stop" button. Double click on the "Stop" button and type in the following code:

```
Private Sub Button2_Click(...) Handles Button2.Click
    End
End Sub
```

Double click on the "Go!" button and type in the following code:

```
Private Sub Button1_Click(...) Handles Button1.Click

    Dim sr As IO.StreamReader = IO.File.OpenText("C:\temp\name.txt")
    Dim Line, Text(), lastName, firstName, birth As String

    Line = sr.ReadLine
    Text = Line.Split(" ")
    lastName = Text(0)
    firstName = Text(1)
    birth = Text(2)
    ListBox1.Items.Add(firstName & " " & lastName & " " & ", Born: " & birth)

    Line = sr.ReadLine
    Text = Line.Split(" ")
    lastName = Text(0)
    firstName = Text(1)
    birth = Text(2)
    ListBox1.Items.Add(firstName & " " & lastName & " " & ", Born: " & birth)

End Sub
```

Now, test your code. You should get something that looks like the following.



Let's talk about some of the code here:

```
Dim sr As IO.StreamReader = IO.File.OpenText("C:\temp\name.txt")
```

Variable “sr”:

“sr” is declared to be an “IO” object that is capable of receiving input to your program (it can “read streams” of input).

“sr” is assigned to the “IO” object that points to the file “C:\temp\name.txt”.

Input File Directory Path and Name:

“C:\temp\CE311K\InputOutput\ ” is the directory path and “name.txt“ is the name of the file to be opened. In this example, the path is “hardcoded” into the program. Be sure that the path is correct for the location of your file “name.txt”.

Dim:

```
Dim Line, Text(), lastName, firstName, birth As String
```

The variables “Line”, “Text”, “lastName”, “firstName”, and “birth” are declared to be “string” variables. “Text” is somewhat special as you can tell by the parentheses “()” after the name. This means that it is an “array” variable and can be broken into pieces; we’ll cut it into three pieces, “Text(0)”, “Text(1)”, and “Text(2)”.

ReadLine:

```
Line = sr.ReadLine
```

This line reads one line from the file assigned to the variable “sr” and assigns that string to the variable “Line”.

Split:

```
Text = Line.Split(" ")
```

This line “splits” the variable “Line” into pieces. The splits are made at locations in the string where the character in the double quotes appear, a blank space in this case “ ”. The pieces are assigned to the variable “Text”.

Now that we have split the line we read into pieces, we can assign the pieces to variables:

```
lastName = Text(0)  
firstName = Text(1)  
birth = Text(2)
```

Here we put the first piece “Text(0)” into the variable “lastName”, and so forth.

Output:

```
ListBox1.Items.Add(firstName & " " & lastName & " " & ", Born: " & birth)
```

This adds the indicated output to the listbox “ListBox1”

Write Data to an Output File

Next, let's write the output to another file.

```
Private Sub Button1_Click(...) Handles Button1.Click

    Dim sr As IO.StreamReader = IO.File.OpenText("C:\temp\name.txt")
    Dim sw As IO.StreamWriter = IO.File.CreateText("C:\temp\output.txt")
    Dim Line, Text(), lastName, firstName, birth As String

    sw.WriteLine("This is the output file.")

    Line = sr.ReadLine
    Text = Line.Split(" ")
    lastName = Text(0)
    firstName = Text(1)
    birth = Text(2)
    ListBox1.Items.Add(firstName & " " & lastName & " " & ", Born: " & birth)
    sw.WriteLine(firstName & " " & lastName & " " & ", Born: " & birth)

    Line = sr.ReadLine
    Text = Line.Split(" ")
    lastName = Text(0)
    firstName = Text(1)
    birth = Text(2)
    ListBox1.Items.Add(firstName & " " & lastName & " " & ", Born: " & birth)
    sw.WriteLine(firstName & " " & lastName & " " & ", Born: " & birth)
    sw.Close()

End Sub
```

Let's talk about some of the code here:

```
Dim sw As IO.StreamWriter = IO.File.CreateText("C:\temp\output.txt")
```

Variable “sw”:

“sw” is declared to be an “IO” object that is capable of receiving output from your program (it can “write streams” of output).

“sw” is assigned to the “IO” object that points to the file “C:\temp\output.txt”.

WriteLine:

```
sw.WriteLine("This is the output file")
```

This line writes one line into the file assigned to the variable "sw".

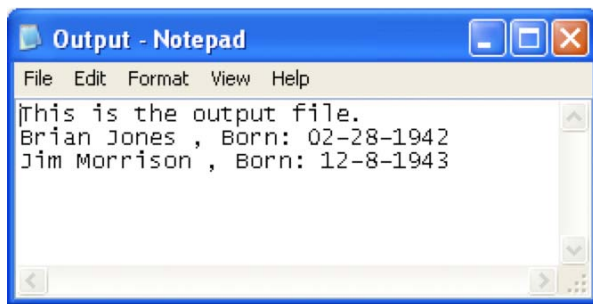
```
sw.WriteLine(firstName & " " & lastName & " " & ", Born: " & birth)
```

This line writes one line into the output file with the variables indicated.

Close:

```
sw.Close()
```

This line closes the output file and allows us to look at the file contents after the program is stopped.



Assignment

Write a VB code with graphic user interface to find the average annual temperature in Austin. Use the program “Notebook” to create a text file to use as input for your program. The input file should contain the information in the following table: one column for the name of the month and one column for the temperature of that month. Your program should open the input file and read it one line at a time and compute the average of the monthly temperatures to get the annual average temperature. Write the result to an output file.

| Month | Temperature (deg F) |
|-----------|---------------------|
| January | 58 |
| February | 63 |
| March | 71 |
| April | 79 |
| May | 84 |
| June | 91 |
| July | 95 |
| August | 95 |
| September | 90 |
| October | 82 |
| November | 71 |
| December | 62 |

Turn in:

1. A printout of the input file from your program.
2. A printout of the VB code used in your program.
3. A printout of the output file from your program.